大規模疎行列の正定値性の保証法 (計算科学の基盤技術としての高速アルゴリズムとその周辺)

AUTHOR(S):

荻田, 武史; Rump, Siegfried M.; 大石, 進一

CITATION:

荻田, 武史 ...[et al]. 大規模疎行列の正定値性の保証法 (計算科学の基盤技術としての高速アルゴリズムとその周辺). 数理解析研究所講究録 2008, 1614: 34-39

ISSUE DATE:

2008-10

URL:

http://hdl.handle.net/2433/140115

数理解析研究所講究録
第1614巻 2008年 34-39

大規模疎行列の正定値性の保証法

荻田 武史 *,‡　　Siegfried M. Rump†,‡　　大石 進一 ‡

* 東京女子大学 文理学部 数理学科
† Institute for Reliable Computing, Hamburg University of Technology
‡ 早稲田大学 理工学術院

1 はじめに

本論文は，大規模で疎な対称行列の正定値性の保証法について考える．行列の正定値性の保証は，たとえば A が実対称行列であるが，正定値であることがわからない場合等にそれを証明し，連立一次方程式の近似解の誤差限界を計算することに応用できる [7].

近年，Rump はコレスキー分解に基づく正定値性の高速な保証法 [6] を提案した．その成果は，Matlab のツールボックスである INTLAB [3] に，関数 `isspd` として実装済みとなっている．この Rump の方式のポイントは，事前誤差評価によって適当な α を求めた後は，$A-\alpha I$ に対する浮動小数点演算によるコレスキー分解が成功するかどうかだけで正定値性が判定できるところにある．

本論文では，Rump の方式を応用し，ブロックコレスキー分解を用いてより大きな問題を取り扱える方式を開発することである．

以下，$\mathbb{F}$ を浮動小数点数の集合とする．また，$\mathbf{u}$ を浮動小数点演算の相対精度（IEEE 754 倍精度なら $\mathbf{u}=2^{-53}$）とし，$\gamma_k := k\mathbf{u}/(1-k\mathbf{u})$ と定義する．

2 コレスキー分解

便宜上，コレスキー分解のアルゴリズムを示しておく．$B=(b_{ij})\in\mathbb{R}^{n\times n}$，$B=B^T$ とする．次のアルゴリズムは，$B=GG^T$ となるような B のコレスキー分解を実行する．ただし，$G=(g_{ij})\in\mathbb{R}^{n\times n}$ は下三角行列である．

アルゴリズム 2.1 コレスキー分解.

$$
\begin{aligned}
&\textit{for } i=1:n\\
&\quad \textit{for } j=1:i-1\\
&\qquad g_{ij}=(b_{ij}-\sum_{k=1}^{j-1} g_{ik}g_{jk})/g_{jj}\\
&\quad \textit{end}\\
&\quad g_{ii}=(b_{ii}-\sum_{k=1}^{i-1} g_{ik}^2)^{1/2}\\
&\textit{end}
\end{aligned}
$$

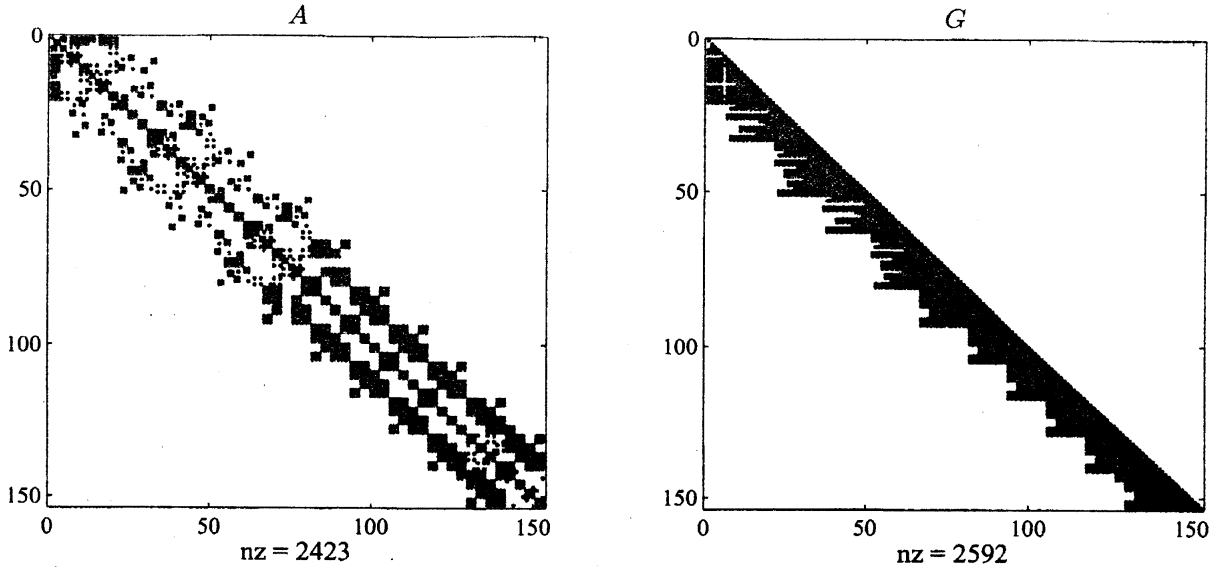

図 1: コレスキー分解における fill-in の例.

もし，B が正定値であるならば，(丸め誤差がなければ) アルゴリズム 2.1 は成功裏に完了する (逆もまた同様)．ここで,「成功裏に完了」とは，コレスキー分解のプロセスで負の数の平方根が現れないことを意味する.

実際には，浮動小数点演算を用いた場合は丸め誤差が発生するため，行列 A の正定値が正定値であるかどうかを知りたいときに，A のコレスキー分解が成功しても，必ずしも A が正定値であることは保証できない.

これまで知られている正定値性の保証法 [4, 5] は，いずれも A の対角要素をシフトした行列 $B := A - \alpha I$ の浮動小数点演算によるコレスキー分解を用いている．疎行列にコレスキー分解を用いると大量の fill-in が起きるが (図 1 参照)，そのようなアプローチでも，B のスパース性がある程度は保たれるため，それなりに大規模な問題 (具体的な問題サイズは計算環境に依存するが，連立一次方程式に対して直接解法で近似解を得ることができるような範囲の問題) を扱うことができるが，それより大きな問題に対しては，主にメモリ量の問題から適用できない.

3 Rump の方式

まず，Rump による下記の定理 (たとえば，[6, 7]) を示す．これは，Demmel による定理 [1] の改良版である.

定理 3.1 対称行列 $B = (b_{ij}) \in \mathbb{F}^{n\times n}$ (ただし，$b_{jj} \geq 0$) に対し，浮動小数点演算によってアルゴリズム 2.1 を適用したとする．また，$\varphi_k := \gamma_k(1-\gamma_k)^{-1}$ とおく．このとき，オーバフローやアンダフローを除外すると，以下が成立する：

i) $\lambda_{\min}(B) \geq \sum_{j=1}^{n} \varphi_{j+1} b_{jj}$ であれば，アルゴリズム 2.1 は成功裏に完了する.

ii) $\lambda_{\min}(B) < -\sum_{j=1}^{n} \varphi_{j+1} b_{jj}$ であれば，アルゴリズム 2.1 は計算過程で負の数の平方根が現れ，途中で終了する．

定理 3.1 の ii) の対偶を考えると，アルゴリズム 2.1 が成功裏に完了したとき

$$\lambda_{\min}(B) \geq -\sum_{j=1}^{n} \varphi_{j+1} b_{jj} \tag{1}$$

であることが言える．ここで，$\beta_1 \geq \sum_{j=1}^{n} \varphi_{j+1} a_{jj}$ とし，$\beta_2 > \beta_1$ に対して $B := A - \beta_2 I$ と決める．もし，B の浮動小数点演算によるコレスキー分解が途中で終了した場合，A の正定値性は保証できない．仮に，これが成功裏に完了したとすると，丸め誤差のため B が正定値かどうかはわからないが，式 (1) から下記は成り立つ：

$$\begin{aligned}
\lambda_{\min}(A) - \beta_2 &= \lambda_{\min}(A - \beta_2 I) = \lambda_{\min}(B) \\
&\geq -\sum_{j=1}^{n} \varphi_{j+1} b_{jj} = -\sum_{j=1}^{n} \varphi_{j+1}(a_{jj} - \beta_2) \\
&\geq -\sum_{j=1}^{n} \varphi_{j+1} a_{jj} \geq -\beta_1
\end{aligned}$$

よって

$$\lambda_{\min}(A) \geq \beta_2 - \beta_1 > 0$$

となり，A の正定値性が保証される．したがって，正定値性を保証するだけなら，$\beta_2 = \beta_1$ と定めれば良い．

Rump の方式の詳細については，文献 [6] を参照されたい．

4 ブロックコレスキー分解

ブロックコレスキー分解は良く知られた方法である（たとえば，[2]）．A を帯幅が L の実対称 $n \times n$ 帯行列とする．このとき

$$A = \begin{bmatrix} A_{11} & A_{21}^T & O \\ A_{21} & A_{22} & A_{32}^T \\ O & A_{32} & A_{33} \end{bmatrix} \tag{2}$$

と書ける．ただし，$A_{11}, A_{21}, A_{22} \in \mathbb{F}^{L \times L}$，$A_{32} \in \mathbb{F}^{(n-2L) \times L}$ および $A_{33} \in \mathbb{F}^{(n-2L) \times (n-2L)}$ である．

今，コレスキー分解 $A_{11} = G_{11} G_{11}^T$ が得られたとする．ただし，$G_{11} \in \mathbb{F}^{L \times L}$ は下三角行列である．次に，A の部分行列 $A^{(1)}$ を

$$A^{(1)} := \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} \tag{3}$$

とおいたとき

$$A^{(1)} = \begin{bmatrix} G_{11} & O \\ G_{21} & F \end{bmatrix} \begin{bmatrix} G_{11}^T & G_{21}^T \\ O & F^T \end{bmatrix} \tag{4}$$

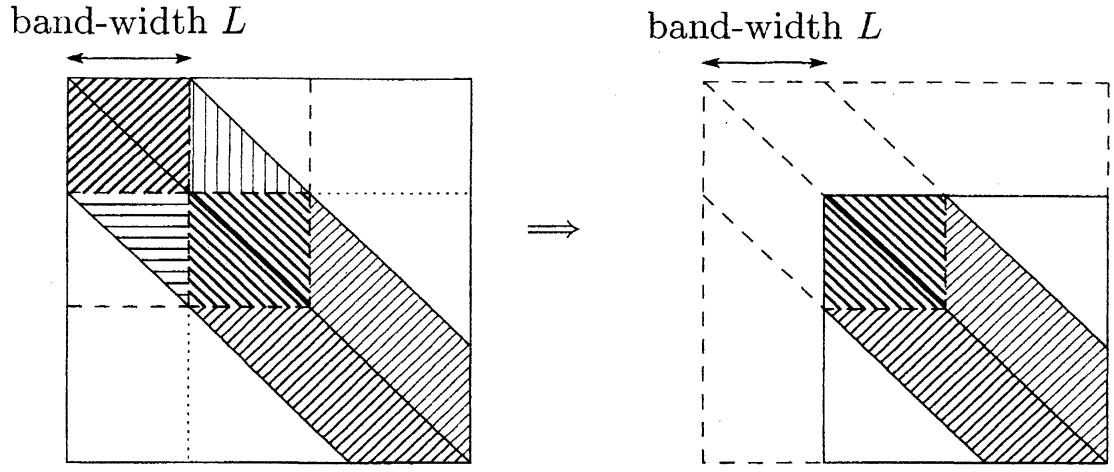

図 2: ブロックコレスキー分解.

のように分解できたとする．ただし，$G_{21} \in \mathbb{F}^{(n-2L)\times L}$ および $F \in \mathbb{F}^{(n-2L)\times(n-2L)}$ である．このとき，下記が成り立つ：

$$A_{11} = G_{11}G_{11}^T \tag{5}$$

$$A_{21} = G_{21}G_{11}^T \tag{6}$$

$$A_{22} = G_{21}G_{21}^T + FF^T \tag{7}$$

式 (6) から，行列方程式 $G_{21}G_{11}^T = A_{21}$ を解くことによって G_{21} を得ることができる．さらに，$B := A_{22} - G_{21}G_{21}^T$ を計算する．次のステップは $B = FF^T$ の計算だが，これは B のコレスキー分解に他ならない．以下，これまでの操作を繰り返せば良い（図 2 参照）．

Rump の方式による行列 A の正定値性の保証に限ると，B を得た後は，G_{11} と G_{21} の情報は不要になり，$A^{(2)}$ の分解に進むことができる：

$$A^{(2)} = \begin{bmatrix} B & A_{32}^T \\ A_{32} & A_{33} \end{bmatrix} \tag{8}$$

したがって，$A_{11} = G_{11}G_{11}^T$ を計算するようなコレスキー分解のルーチン，行列方程式 $G_{21}G_{11}^T = A_{21}$ を解くルーチンおよび行列乗算 $G_{21}G_{21}^T$ のルーチンがあれば，上記のブロックコレスキー分解は実行可能であり，多くの場合にそれらの高速なライブラリが利用可能である．

式 (6) の G_{11}，G_{21} および A_{21} の関係から，Rump の方式にブロックコレスキー分解を適用すると，最低限必要なメモリ量は A の帯幅 L に依存し，いくつかの $L \times L$ 行列を格納できれば済むという意味で，Rump の方式に必要なメモリ量を大幅に低減することが可能となる．

密行列のコレスキー分解については，LAPACK や ScaLAPACK 等，並列化も含めた多くの高速なライブラリが利用可能であるため，ブロック行列は，ある程度疎な行列であっても密行列として取り扱う方が効率的な場合もある．その場合，疎行列のオーダリングによって帯幅を最小限にすることが重要となるため，minimum degree ordering 系統のオーダリング手法ではなく，reverse Cuthill-McKee ordering を用いるほうが良いであろう．

ブロックコレスキー分解に基づく提案方式の主計算部分を Matlab 形式で下記に示す．

表 1: 正定値性の保証に要した計算時間.

テスト行列	n	帯幅 オーダリングなし／あり	計算時間（秒）
DNVS/ship_003	121,728	3659/3659	260
DNVS/shipsec1	140,874	5238/5238	538
Rothberg/cfd2	123,440	4333/2179	127
Schenk_AFE/af_shell(3,4,7,8)	504,855	4909/2470	633
GHS_psdef/apache2	715,176	65837/2993	1176

```
index = 1:L;                                % L: bandwidth
G = B(index,index);
for k=1:L:n
    G = chol(G);                            % floating-point Cholesky
    pre_index = index;
    index = min(k+L,n):min(k+2*L-1,n);      % next indeces
    E = G \ B(pre_index,index);             % Solve G*E = B for E
    G = B(index,index) - E'*E;              % Update the next block
end
```

5 数値実験

正定値性の保証について，提案方式の性能を評価する．テスト行列は，University of Florida Sparse Matrix Collection から得た．計算環境は，下記の通りである．

CPU: Intel Dual-Core Xeon 2.80GHz × 4 processors
Memory: 32GB
OS: Red Hat Enterprise Linux WS
Software: Matlab Version 7.1.0.183 (R14) Service Pack 3

行列のオーダリングには，reverse Cuthill-McKee ordering を用いた．

数値実験結果を表 1 に示す．この表に掲載したテスト行列は，INTLAB の `isspd` ではメモリ不足で実行できなかったが，提案方式ではすべて正定値性の保証に成功したものである（この表に掲載していないテスト行列で，正定値性の保証に失敗したものもある）．テスト行列によっては，帯幅が大きく減少するものもあった（たとえば，GHS_psdef/apache2）．これにより，ある程度大規模な問題でも，比較的実用的な範囲で正定値性が保証できる場合があることが確認できた．

6 おわりに

本論文では，大規模な対称疎行列に対し，その正定値性を保証する方法を提案した．提案方式は，Rump の方式にブロックコレスキー分解を適用したものであった．また，数値実験によって，その有効性を確認した．

提案方式も含めて，Rumpの方式が正定値性の保証に失敗した場合でも適用可能な，よりロバストな方式を開発するのが今後の課題である．

参考文献

[1] J. B. Demmel: On floating point errors in Cholesky, LAPACK Working Note 14 CS-89-87, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, 1989.

[2] G. H. Golub, C. F. Van Loan: Matrix Computations, third edition, The Johns Hopkins University Press, Baltimore and London, 1996.

[3] S. M. Rump: INTLAB - INTerval LABoratory, Version 5.5, 2008. `http://www.ti3.tu-harburg.de/~rump/intlab/`

[4] S. M. Rump: Validated solution of large linear systems, Computing, Suppl. 9 (1993), 191–212.

[5] S. M. Rump: Verification methods for dense and sparse systems of equations, Topics in Validated Computations – Studies in Computational Mathematics (J. Herzberger ed.), Elsevier, Amsterdam, 63–136, 1994.

[6] S. M. Rump: Verification of positive definiteness, BIT Numerical Mathematics, 46 (2006), 433–452.

[7] S. M. Rump, T. Ogita: Super-fast validated solution of linear systems, J. Comp. Appl. Math., 199:2 (2007), 199–206.